

# This bears a striking resample-lance

SAMPLING IN R



**Richie Cotton**

Curriculum Architect at DataCamp

# With or without

Sampling without replacement



Sampling with replacement ("resampling")



# Simple random sampling without replacement

Population



Sample





# Simple random sampling with replacement

Population



Sample



# Why sample with replacement?

- Think of the `coffee_ratings` data as being a sample of a larger population of all coffees.
- Think about each coffee in our sample as being representative of many different coffees that we don't have in our sample, but do exist in the population.
- Sampling with replacement is a proxy for including different members of these groups in our sample.

# Coffee data preparation

```
coffee_focus <- coffee_ratings %>%  
  select(variety, country_of_origin, flavor) %>%  
  rowid_to_column()
```

```
glimpse(coffee_focus)
```

```
Rows: 1,338  
Columns: 4  
$ rowid      <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, ...  
$ variety    <chr> NA, "Other", "Bourbon", NA, "Other", NA, "Other", N...  
$ country_of_origin <chr> "Ethiopia", "Ethiopia", "Guatemala", "Ethiopia", "E...  
$ flavor     <dbl> 8.83, 8.67, 8.50, 8.58, 8.50, 8.42, 8.50, 8.33, 8.6...
```

# Resampling with slice\_sample()

```
coffee_resamp <- coffee_focus %>%  
  slice_sample(prop = 1, replace = TRUE)
```

```
# A tibble: 1,338 x 4  
  rowid variety country_of_origin flavor  
  <int> <chr>      <chr>          <dbl>  
1  1253 Bourbon    Guatemala      6.92  
2   186 Caturra   Colombia      7.58  
3  1185 Bourbon    Guatemala      7.42  
4  1273 NA         Philippines     6.5  
5  1042 Caturra   Honduras      7.33  
6   195 Caturra   Guatemala      7.75  
7  1219 Typica     Mexico         7  
8   952 Caturra   Honduras      7.5  
9    41 Caturra   Thailand      8.33  
10  460 Caturra   Honduras      7.67  
# ... with 1,328 more rows
```

# Repeated coffees

```
coffee_resamp %>%  
  count(rowid, sort = TRUE)
```

```
# A tibble: 844 x 2  
  rowid      n  
  <int> <int>  
1   704     5  
2   913     5  
3  1070     5  
4    16     4  
5   180     4  
6   230     4  
7   234     4  
8   342     4  
9   354     4  
10  423     4  
# ... with 834 more rows
```



# Missing coffees

```
coffee_resamp %>%  
  summarize(  
    coffees_included = n_distinct(rowid),  
    coffees_not_included = n() - coffees_included  
  )
```

```
# A tibble: 1 x 2  
  coffees_included coffees_not_included  
          <int>          <int>  
1             844             494
```

# Bootstrapping

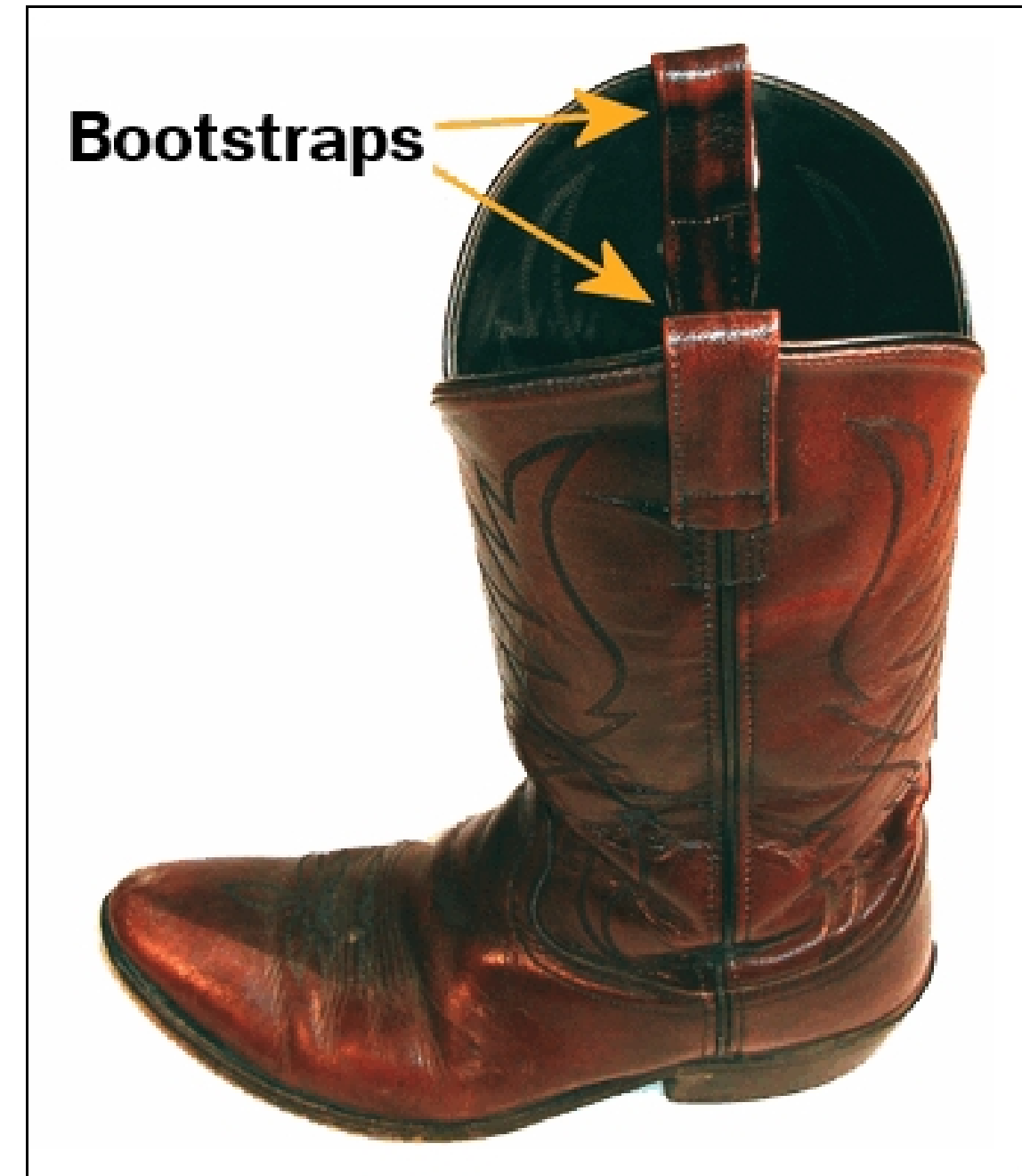
The opposite of sampling from a population.

*Sampling:* going from a population to a smaller sample.

*Bootstrapping:* building up a theoretical population from your sample.

Bootstrapping use case

- Develop understanding of sampling variability using a single sample.



# Bootstrapping process

1. Make a resample of the same size as the original sample.
2. Calculate the statistic of interest for this bootstrap sample.
3. Repeat steps 1 and 2 many times.

The resulting statistics are called *bootstrap statistics* and when viewed to see their variability a *bootstrap distribution*.

# Bootstrapping coffee mean flavor

```
# Step 3. Repeat many times
```

```
mean_flavors_1000 <- replicate(  
  n = 1000,  
  expr = {
```

```
    coffee_focus %>%
```

```
      # Step 1. Resample
```

```
      slice_sample(prop = 1, replace = TRUE) %>%
```

```
        # Step 2. Calculate statistic
```

```
        summarize(mean_flavor = mean(flavor, na.rm = TRUE)) %>%
```

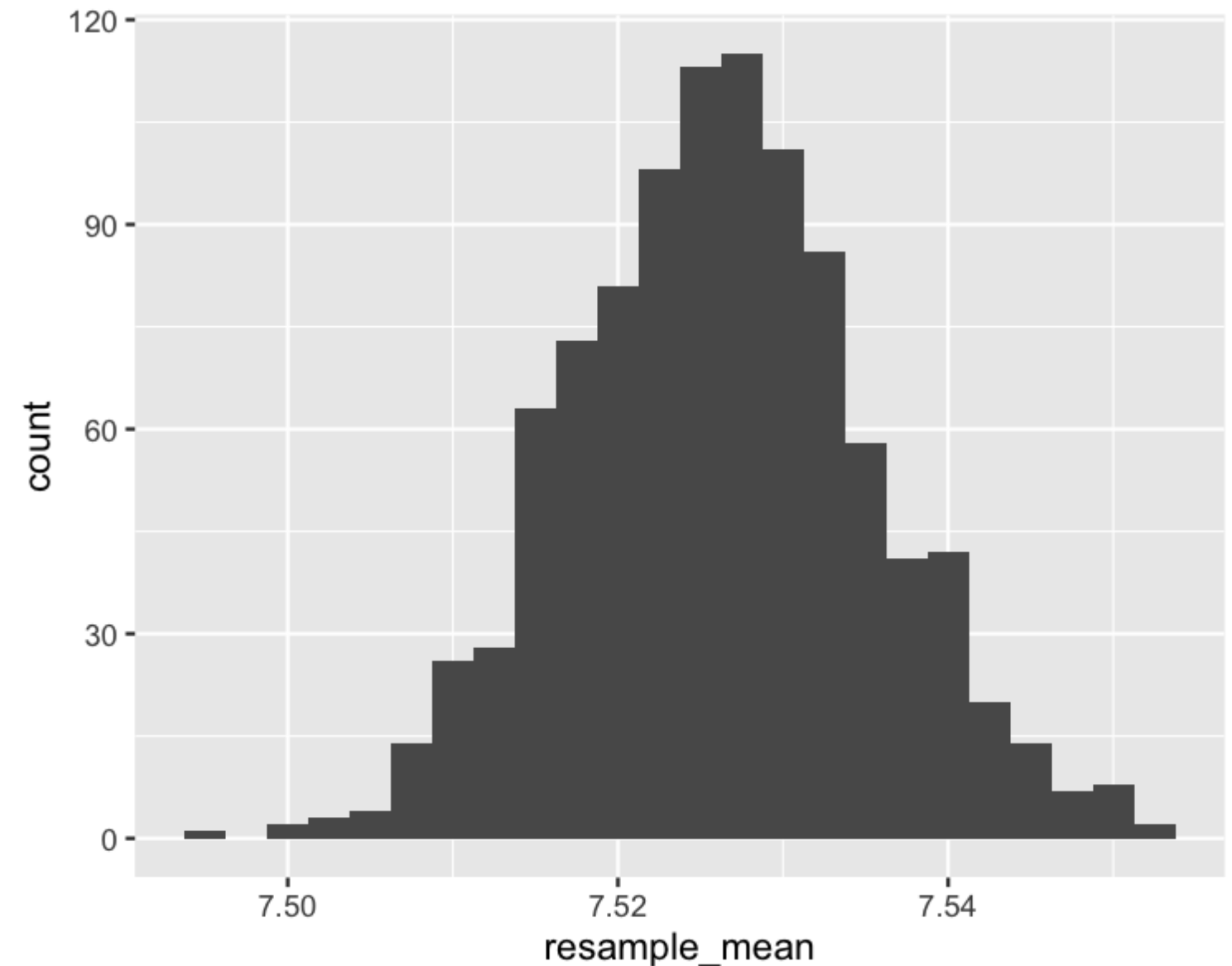
```
        pull(mean_flavor)
```

```
  })
```

# Bootstrap distribution histogram

```
bootstrap_distn <- tibble(  
  resample_mean = mean_flavors_1000  
)
```

```
ggplot(bootstrap_distn, aes(resample_mean)) +  
  geom_histogram(binwidth = 0.0025)
```





**Let's practice!**  
SAMPLING IN R

# A breath of fresh error

SAMPLING IN R



**Richie Cotton**

Curriculum Architect at DataCamp

# Coffee focused subset

```
set.seed(19790801)
coffee_sample <- coffee_ratings %>%
  select(variety, country_of_origin, flavor) %>%
  rowid_to_column() %>%
  slice_sample(n = 500)
glimpse(coffee_sample)
```

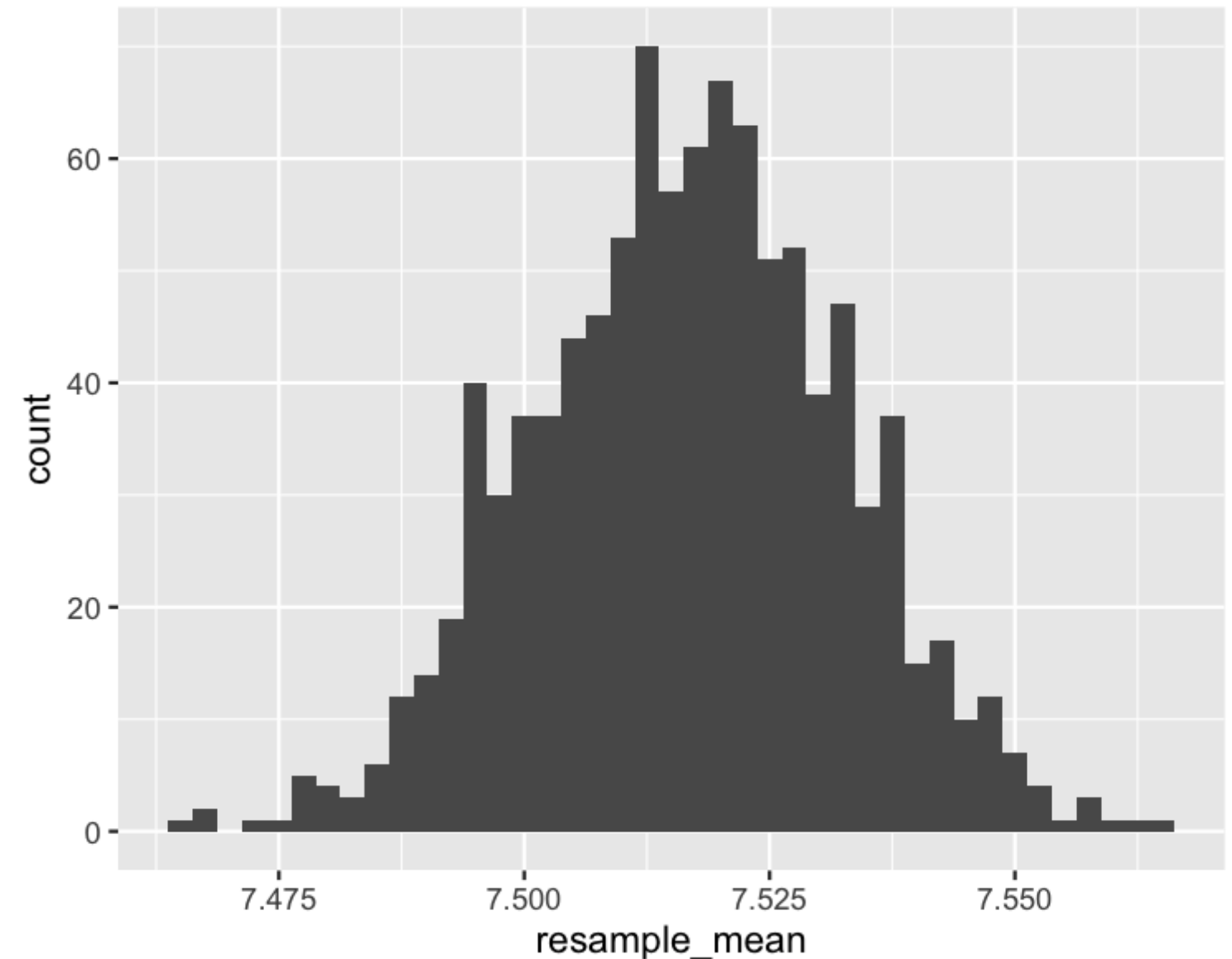
```
Rows: 500
Columns: 4
$ rowid      <int> 10, 278, 458, 622, 131, 385, 1292, 47, 904, 1020, 5...
$ variety    <chr> "Other", "Bourbon", NA, "Caturra", "Caturra", "Yell...
$ country_of_origin <chr> "Ethiopia", "Guatemala", "Colombia", "Thailand", "C...
$ flavor     <dbl> 8.58, 7.75, 7.75, 7.50, 8.00, 7.83, 7.17, 8.08, 7.3...
```

# The bootstrap of mean coffee flavors

```
mean_flavors_1000 <- replicate(  
  n = 1000,  
  expr = coffee_sample %>%  
    slice_sample(prop = 1, replace = TRUE) %>%  
    summarize(mean_flavor = mean(flavor, na.rm = TRUE)) %>%  
    pull(mean_flavor)  
)  
bootstrap_distn <- tibble(  
  resample_mean = mean_flavors_1000  
)
```

# Mean flavor bootstrap distribution

```
ggplot(bootstrap_distn, aes(resample_mean)) +  
  geom_histogram(binwidth = 0.0025)
```





# Sample, bootstrap distribution, population means

## Sample mean

```
coffee_sample %>%  
  summarize(mean_flavor = mean(flavor)) %>%  
  pull(mean_flavor)
```

7.5163

## Estimated population mean

```
bootstrap_distn %>%  
  summarize(mean_mean_flavor = mean(resample_mean)) %>%  
  pull(mean_flavor)
```

7.5167

## True population mean

```
coffee_ratings %>%  
  summarize(mean_mean_flavor = mean(resample_mean)) %>%  
  pull(mean_flavor)
```

7.5260

# Interpreting the means

- The bootstrap distribution mean is usually almost identical to the sample mean.
- It may not be a good estimate of the population mean.
- Bootstrapping cannot correct biases due to differences between your sample and the population.

# Sample sd vs bootstrap distribution sd

Sample standard deviation

```
coffee_focus %>%  
  summarize(sd_flavor = sd(flavor)) %>%  
  pull(sd_flavor)
```

0.3525

Estimated population standard deviation?

```
bootstrap_distn %>%  
  summarize(sd_mean_flavor = sd(resample_mean)) %>%  
  pull(sd_mean_flavor)
```

0.01572

# Sample, bootstrap dist'n, pop'n standard deviations

## Sample standard deviation

```
coffee_focus %>%  
  summarize(sd_flavor = sd(flavor)) %>%  
  pull(sd_flavor)
```

```
0.3525
```

## True standard deviation

```
coffee_ratings %>%  
  summarize(sd_flavor = sd(flavor)) %>%  
  pull(sd_flavor)
```

```
0.3414
```

## Estimated population standard deviation

```
standard_error <- bootstrap_distn %>%  
  summarize(sd_mean_flavor = sd(resample_mean)) %>%  
  pull(sd_mean_flavor)
```

```
standard_error * sqrt(500)
```

```
0.3515
```

*Standard error* is the standard deviation of the statistic of interest.

*Standard error* times square root of sample size estimates the population standard deviation.

# Interpreting the standard errors

- *Estimated standard error* is the standard deviation of the bootstrap distribution for a sample statistic.
- The bootstrap distribution standard error times the square root of the sample size estimates the standard deviation in the population.



**Let's practice!**  
SAMPLING IN R

# Venus infers

SAMPLING IN R



**Richie Cotton**

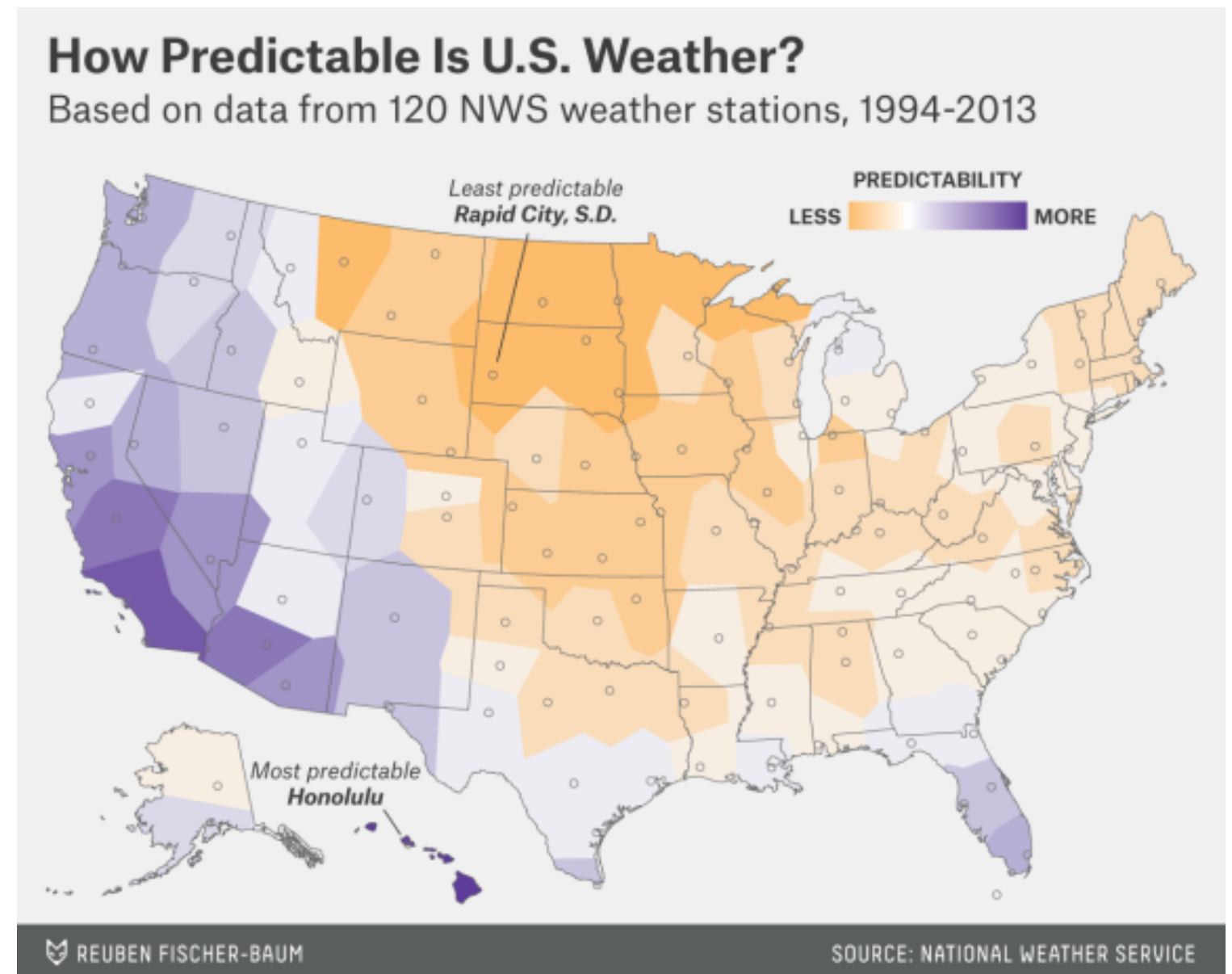
Curriculum Architect at DataCamp

# Confidence intervals

- "Values within one standard deviation of the mean" includes a large number of values from each of these distributions.
- We'll define a related concept called a *confidence interval*.

# Predicting the weather

- Rapid City, South Dakota in the United States has the least predictable weather.
- Your job is to predict the high temperature there tomorrow.



# Your weather prediction

- point estimate = 47 °F (8.3 °C)
- range of plausible high temperature values = 40 to 54 °F (4.4 to 12.8 °C)

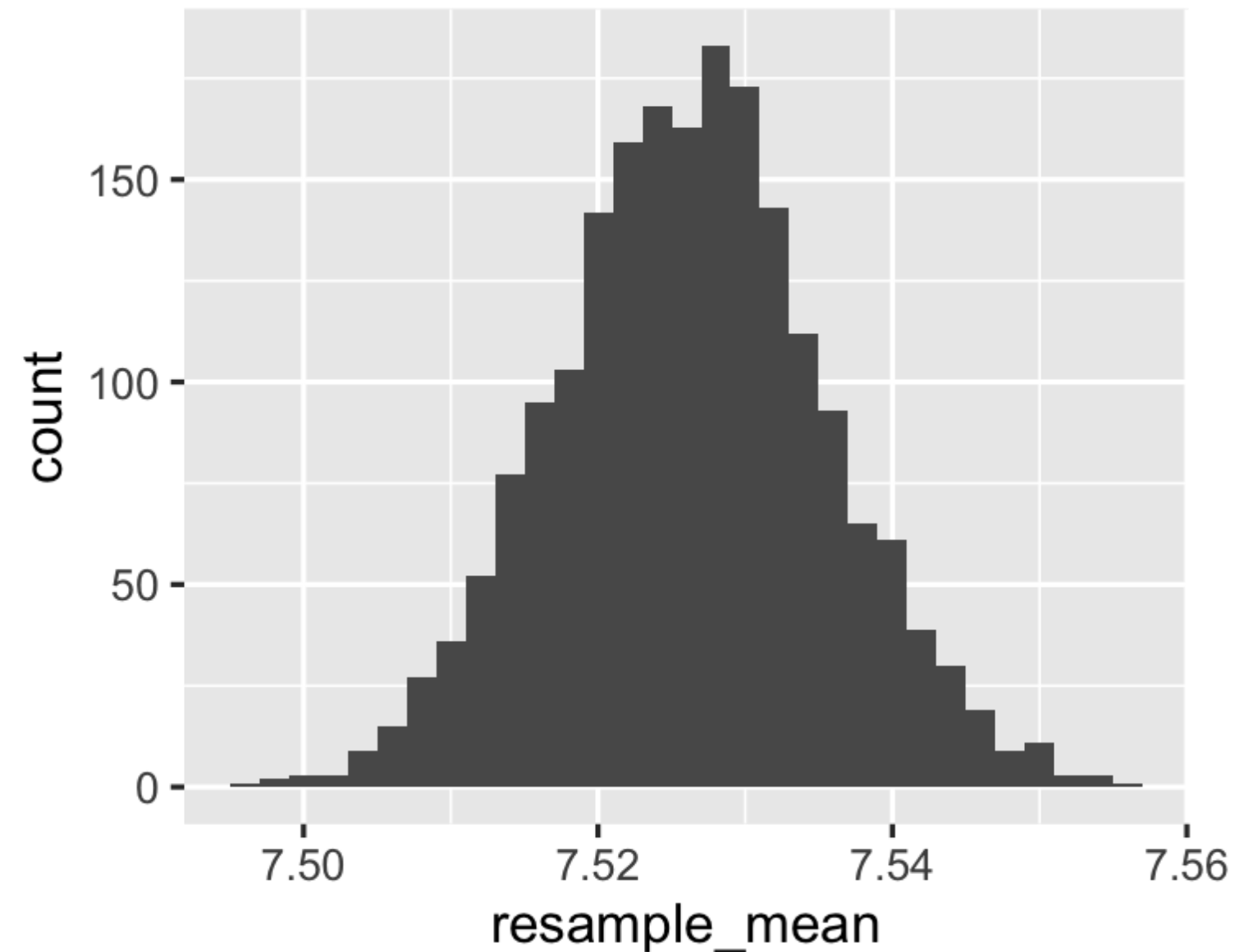


# You just reported a confidence interval

- 40 to 54 °F is a *confidence interval*
- Sometimes written as 47 °F (40 °F, 54 °F) or 47 °F [40 °F, 54 °F]
- ... or,  $47 \pm 7$  °F
- 7 °F is the *margin of error*

# Bootstrap distribution of mean flavor

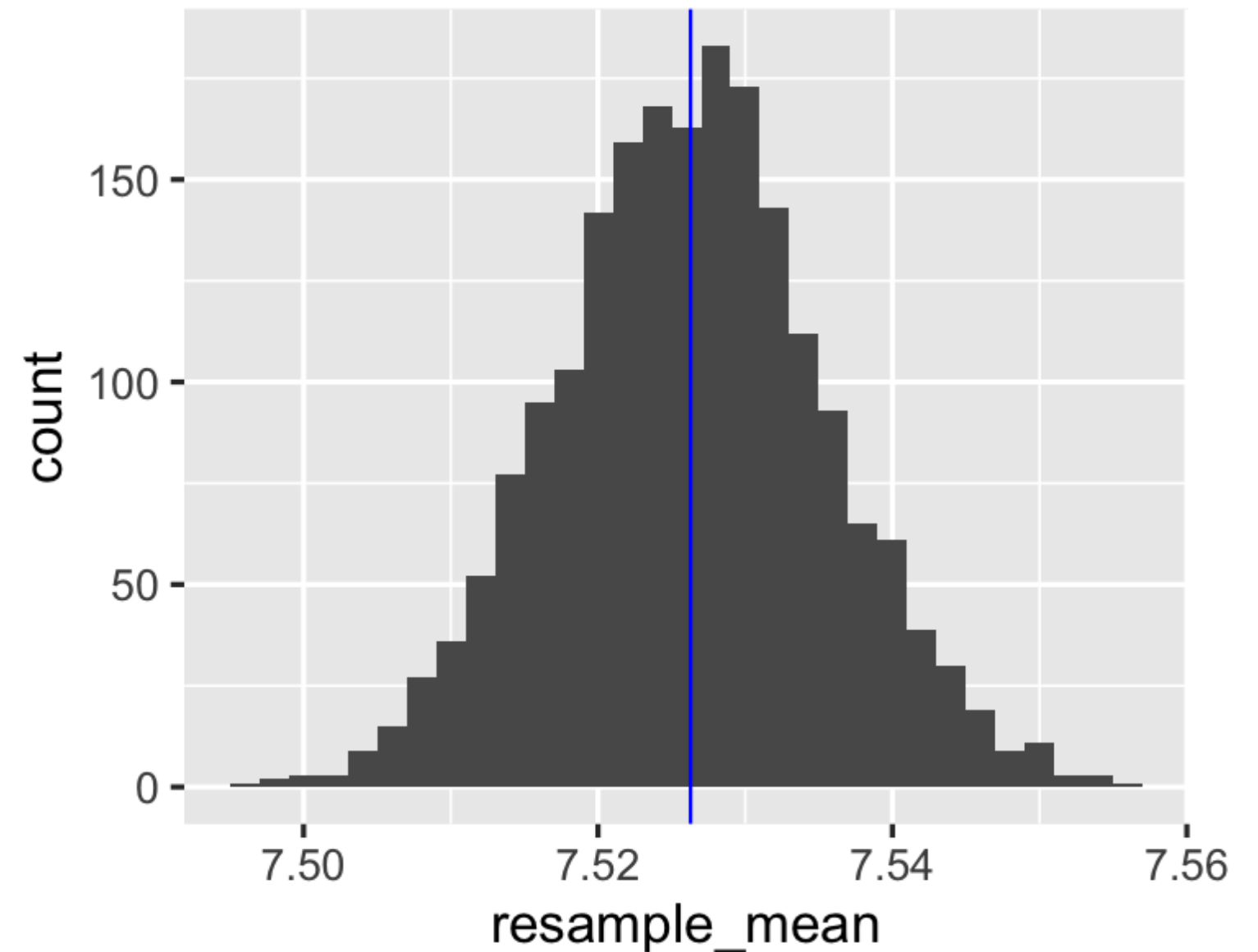
```
ggplot(coffee_boot_distn, aes(resample_mean)) +  
  geom_histogram(binwidth = 0.002)
```



# Mean of the resamples

```
coffee_boot_distn %>%  
  summarize(  
    mean_resample_mean = mean(resample_mean)  
  )
```

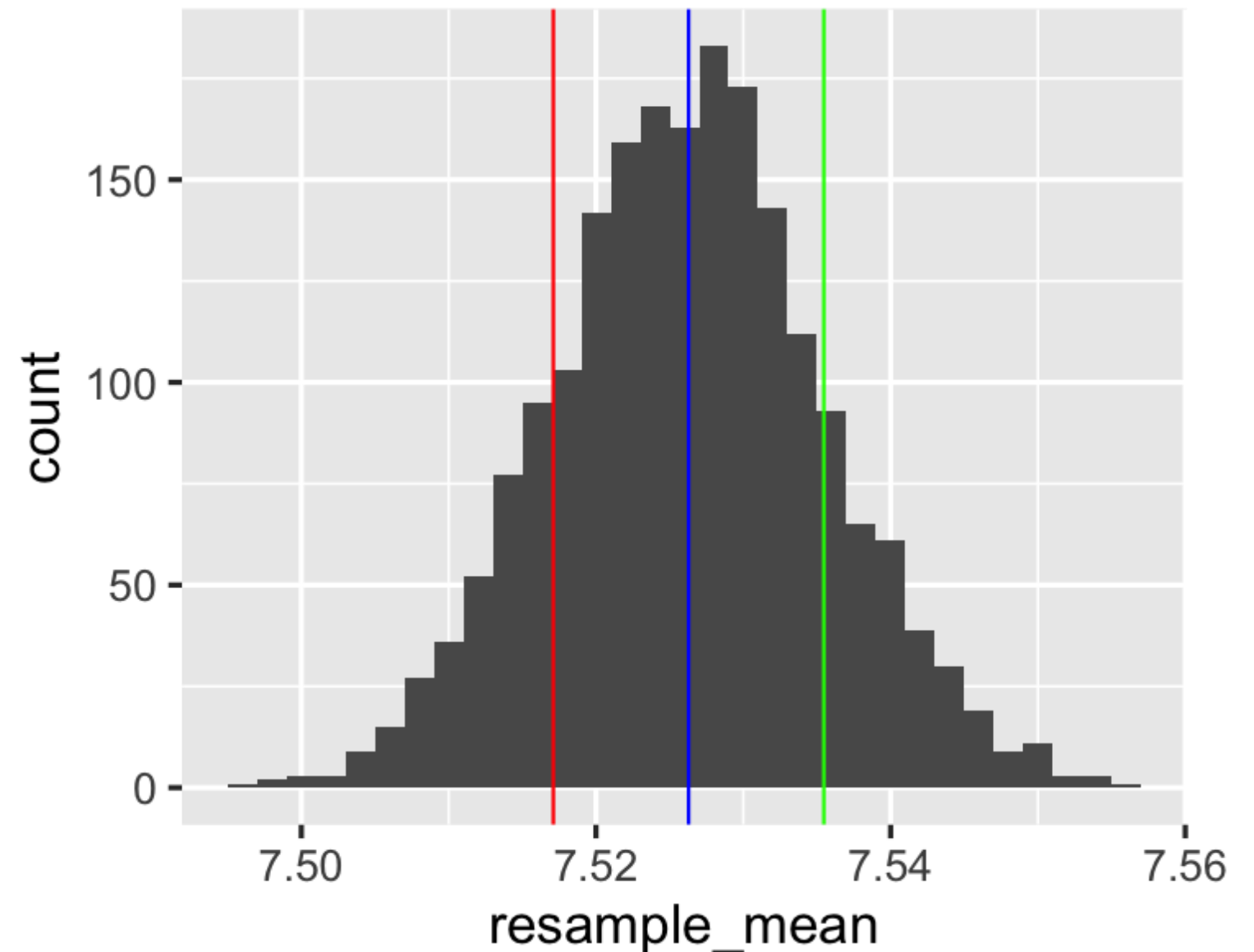
```
# A tibble: 1 x 1  
  mean_resample_mean  
            <dbl>  
1             7.5263
```



# Mean plus or minus one standard deviation

```
coffee_boot_distn %>%  
  summarize(  
    mean_resample_mean = mean(resample_mean),  
    mean_minus_1sd = mean_resample_mean - sd(resample_mean),  
    mean_plus_1sd = mean_resample_mean + sd(resample_mean)  
  )
```

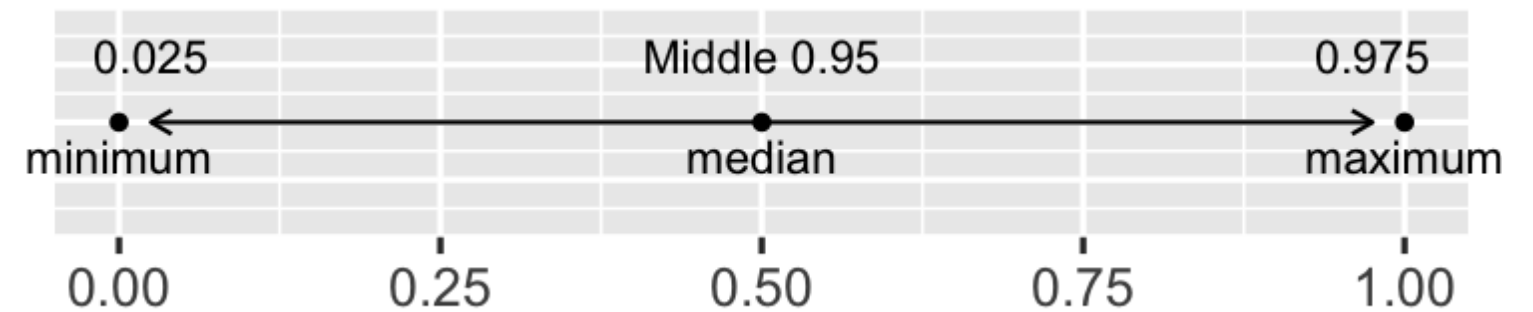
```
# A tibble: 1 x 3  
  mean_resample_mean mean_plus_1sd mean_minus_1sd  
    <dbl>          <dbl>          <dbl>  
1      7.5263      7.5355      7.5171
```



# Quantile method for confidence intervals

```
coffee_boot_distn %>%  
  summarize(  
    lower = quantile(resample_mean, 0.025),  
    upper = quantile(resample_mean, 0.975)  
  )
```

```
# A tibble: 1 x 2  
  lower upper  
  <dbl> <dbl>  
1 7.5087 7.5447
```

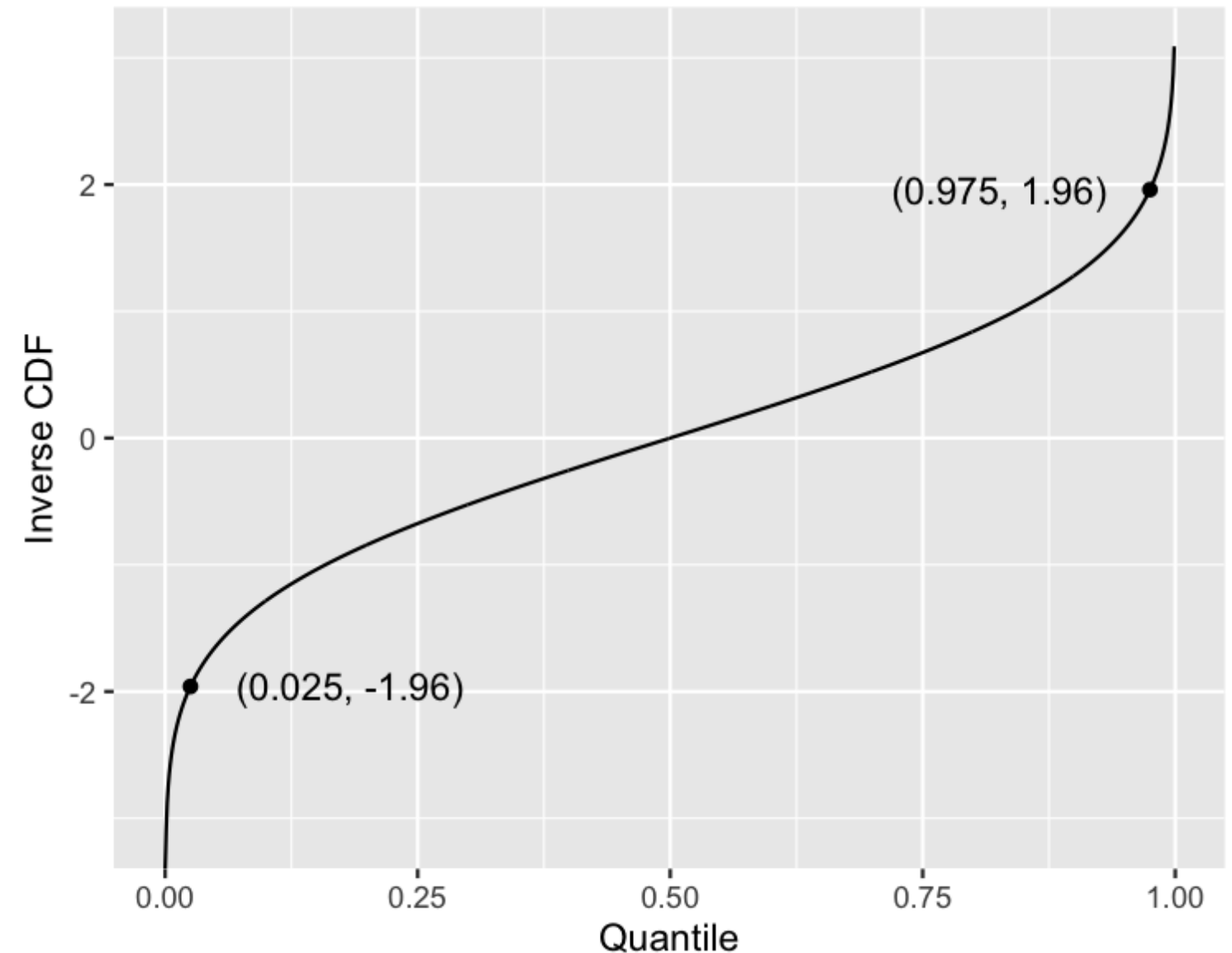


# Inverse cumulative distribution function

- PDF: The bell curve
- CDF: integrate to get area under bell curve
- Inv. CDF: flip x and y axes

```
normal_inv_cdf <- tibble(  
  p = seq(-0.001, 0.999, 0.001),  
  inv_cdf = qnorm(p)  
)
```

```
ggplot(normal_inv_cdf, aes(p, inv_cdf)) +  
  geom_line()
```



<sup>1</sup> See "Introduction to Statistics in R", Ch3, "The Normal Distribution"

# Standard error method for confidence interval

```
coffee_boot_distn %>%  
  summarize(  
    point_estimate = mean(resample_mean),  
    std_error = sd(resample_mean),  
    lower = qnorm(0.025, point_estimate, std_error),  
    upper = qnorm(0.975, point_estimate, std_error)  
  )
```

```
# A tibble: 1 x 4  
  point_estimate std_error lower upper  
      <dbl>      <dbl> <dbl> <dbl>  
1      7.5263 0.0091815 7.5083 7.5443
```

**Let's practice!**  
SAMPLING IN R



# Congratulations

SAMPLING IN R



**Richie Cotton**

Curriculum Architect at DataCamp

# Recap

## Chapter 1

- Sampling basics
- Selection bias
- Pseudo-random sampling

## Chapter 2

- Simple random sampling
- Systematic sampling
- Stratified sampling
- Cluster sampling

## Chapter 3

- Sample size and population parameters
- Creating sampling distributions
- Approximate vs. actual sampling dist'ns
- Central limit theorem

## Chapter 4

- Bootstrapping from a single sample
- Standard error
- Confidence intervals

# The most important things

- The standard deviation of the sampling distribution (a.k.a. the standard error) of a statistic is well-approximated by the standard deviation of the bootstrap distribution of a statistic.
- When calculating confidence intervals, it's OK to assume that bootstrap distributions are approximately normally distributed.

# What's next?

- [Analyzing Survey Data in R](#) and [Survey and Measurement Development in R](#)
- [Experimental Design in R](#) and [A/B Testing in R](#)
- [Foundations of Inference in R](#)
- [Foundation of Probability in R](#) and [Fundamentals of Bayesian Data Analysis in R](#)

**Let's practice!**  
SAMPLING IN R